WEST Search History

| Hide Items | Restore | Clear | Cancel |

DATE: Tuesday, September 28, 2004

Hide?	Set Name	Query	Hit Count
		DB=USPT; PLUR=YES; OP=ADJ	
☐	L21	l2 and l17	0
☐	L20	709/201.ccls.	1074
☐	L19	L17 and simulat$6	5
☐	L18	L17 and l1	13
☐	L17	L16 and l13	46
☐	L16	dynamic$6 near5 thread	395
☐	L15	static$6 near3 thread	174
☐	L14	simulat$6 same dynamic$ near5 thread same static$6 near3 thread	0
☐	L13	static$6 near5 thread	319
☐	L12	static$6 allocated near5 thread	0
☐	L11	static$6allocated near5 thread	0
☐	L10	thread block same (stor$6 or maintain$6) same linked list	3
☐	L9	static$6allocated thread	0
☐	L8	simulat$4 same static$6allocated thread	0
☐	L7	simulat$4 same static$6-allocated thread	0
☐	L6	simulat$4 same static$6 ad3 allocated thread	0
☐	L5	simulat$4 same static$6 ad3 allocated thread and dynamic$6 adj3 allocated thread	0
☐	L4	simulat$4 near5 static$6 allocated thread and dynamic$6 allocated thread	0
☐	L3	simulat$4 near5 static$6 allocated thread same dynamic$6 allocated thread	0
☐	L2	709/226.ccls.	910
☐	L1	718/$.ccls.	3059

END OF SEARCH HISTORY

Google™

| statically allocated thread block stored linke | Search | Advanced Search
Preferences |
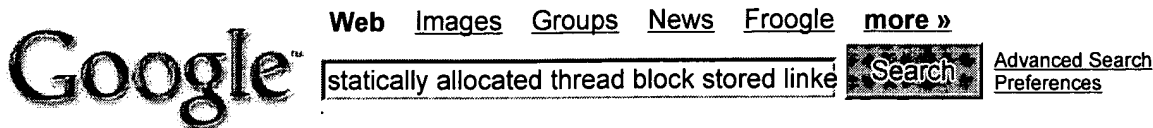
Web   Results **1 - 10** of about **6,540** for <u>statically allocated thread block stored linked list</u>. (**0.24** seconds)

[PDF] <u>7. **Threads** Thread1 Thread2 Thread3</u>
File Format: PDF/Adobe Acrobat - <u>View as HTML</u>
... In particular, there will be three **statically-allocated threads** that are each allowed
to execute 10 ms in a round ... The **Thread** Control **Block** (TCB) will ...
www.ece.utexas.edu/~valvano/EE360P/PDF/Ch7.pdf - <u>Similar pages</u>

[PDF] <u>Lab 18 Real Time Preemptive Multi-Treaded Operating System</u>
File Format: PDF/Adobe Acrobat - <u>View as HTML</u>
... be placed into the tcb of each **thread** (eg, register ... The space for the tcb's is **allocated**
statically and never ... You must be careful to **block** and wakeup the ...
www.ece.utexas.edu/~valvano/manual/Lab18.pdf - <u>Similar pages</u>
[<u>More results from www.ece.utexas.edu</u>]

[PDF] <u>**Thread** Migration in the Presence of Pointers</u>
File Format: PDF/Adobe Acrobat - <u>View as HTML</u>
... The table itself (depicted in Figure) is **statically allocated** at the ... registering
and (b) releasing pointers Finally the **thread** control **block** (tcb) of ...
techreports.larc.nasa.gov/icase/1996/icase-1996-73.pdf - <u>Similar pages</u>

<u>MPI Opaque Objects</u>
... which will need to be **statically** calculable; the ... Elements are **allocated** from these
arrays by using a ... global **linked list**, a special **thread** lock, allocation_lock ...
www-unix.mcs.anl.gov/mpi/mpich/adi3/mpich2/node15.htm - 24k - <u>Cached</u> - <u>Similar pages</u>

<u>ICS 141 Fall 2003, Final Exam Study Guide</u>
... a **thread**-safe Queue class to **block** until there ... site (java.sun.com) has a **thread** tutorial,
though ... language whose local variables are **allocated statically** (ie at ...
www.ics.uci.edu/~thornton/ics141/FinalStudyGuide.html - 21k - <u>Cached</u> - <u>Similar pages</u>

<u>Phase I Specifications</u>
... can be used, based on a **statically allocated** array in ... and data pages to newly **allocated**
user pages ... application which represents the main **thread** of execution of ...
www.cs.tufts.edu/comp/111/phase2/specs.html - 20k - <u>Cached</u> - <u>Similar pages</u>

<u>United States Patent Application: 0040167947</u>
... no longer consider "reachable." **Statically allocated** objects represented by ... and collector
in concurrent execution **threads**. ... faster; newly **allocated** objects tend ...
appft1.uspto.gov/. ../&RS=AN/ - 101k - <u>Cached</u> - <u>Similar pages</u>

<u>BottomHalves - DebianWiki</u>
... Defined in <kernel/softirq.c> & <linux/interrupt.h>; Max 32; **Statically allocated**; ...
The **threads** are each named ksoftirqd/n where n is the processor ... **statically** */: ...
wiki.debian.org.tw/index.php/BottomHalves - 28k - <u>Cached</u> - <u>Similar pages</u>

[PS] <u>Venus Data Structures Lily Mummert v1.0 5/21/97</u>
File Format: Adobe PostScript - <u>View as Text</u>
... VSG's ffl Hoard database ffl Users ffl **Threads** ffl Local ... ffl **statically** bound to
fsobj ffl container o/les used ... ffl range pre-**allocated** o/ds ffl version vector ...
www.coda.cs.cmu.edu/doc/ps/venus-ds.ps.gz - <u>Similar pages</u>

<u>Digital works - TechPage</u>
... on chip in the CPU may be setup **statically**, no page ... when some task writes outside
of its **allocated** memory. ... is checked in the context of the **thread** that mallocs ...
www.digitalworks.iwarp.com/lattur.htm - 101k - <u>Cached</u> - <u>Similar pages</u>

Goooooooooogle ▶

Result Page: **1** <u>2</u> <u>3</u> <u>4</u> <u>5</u> <u>6</u> <u>7</u> <u>8</u> <u>9</u> <u>10</u>   **Next**

Free! Get the Google Toolbar. <u>Download Now</u> - <u>About Toolbar</u>

Google ▾ [▾] | 🔍 Search Web ▾ | 🔲 49 Pop-ups blocked | 📰 News | 📋 AutoFill | ✏️

[statically allocated thread block s] [Search]

<u>Search within results</u> | <u>Language Tools</u> | <u>Search Tips</u> | <u>Dissatisfied? Help us improve</u>

<u>Google Home</u> - <u>Advertising Programs</u> - <u>Business Solutions</u> - <u>About Google</u>

©2004 Google

Static and dynamic partitioning of pointers as links and threads

Full text   📄Pdf (922 KB)

Authors   David S. Wise   Computer Science Department, Indiana University, Bloomington, Indiana
Joshua Walgenbach Computer Science Department, Indiana University, Bloomington, Indiana

Additional Information: abstract   references   citings   index terms   collaborative colleagues

Tools and Actions: Discussions   Find similar Articles   Review this Article
Save this Article to a Binder   Display in BibTex Format

DOI Bookmark: Use this link to bookmark this Article: http://doi.acm.org/10.1145/232627.232634
What is a DOI?

↑ ABSTRACT

Identifying some pointers as invisible threads, for the purposes of storage management, is a generalization from several widely used programming conventions, like threaded trees. The necessary invariant is that nodes that are accessible (without threads) emit threads only to other accessible nodes. Dynamic tagging or static typing of threads ameliorates storage recycling both in functional and imperative languages.We have seen the distinction between threads and links sharpen both hardware- and software-supported storage management in SCHEME, and also in C. Certainly, therefore, implementations of languages that already have abstract management and concrete typing, should detect and use this as a new static type.

↑ REFERENCES

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

1   Andrew W. Appel, Garbage collection can be faster than stack allocation, Information Processing Letters, v.25 n.4, p.275-279, June 17, 1987

2   A. W. Appel & Z. Shao. An empirical and analytic study of stack vs. heap cost for languages with closures. J. Funct. Programming (to appear).

3   Henry G. Baker, Jr., List processing in real time on a serial computer, Communications of the
ACM, v.21 n.4, p.280-294, April 1978

4   Henry G. Baker, Lively linear Lisp: "look ma, no garbage!", ACM SIGPLAN Notices, v.27 n.8, p.89-
98, Aug. 1992

5   D. W. Clark and C. C. Green. A note on shared structure in LmP. Inform. Proc. Ltrs. 7, 6 (October
1978), 312-314.

6   Jacques Cohen, Garbage Collection of Linked Data Structures, ACM Computing Surveys (CSUR),
v.13 n.3, p.341-367, Sept. 1981

7   George E. Collins, A method for overlapping and erasure of lists, Communications of the ACM, v.3
n.12, p.655-657, Dec. 1960

8   L. Peter Deutsch , Daniel G. Bobrow, An efficient, incremental, automatic garbage collector,
Communications of the ACM, v.19 n.9, p.522-526, Sept. 1976

9   D. P. Friedman and D. S. Wise. Reference counting can manage the circular environments of
mutual recursion. Inform. Proc. Ltrs. 8, i (January 1979), 41-44.

10 W. Goldman. The Princess Bride, (screenplay). Nelson Entertainment & Twentieth-Century Fox
Film Corp. (1987), Miracle Max scene.

11 Ellis Horowitz , Sartaj Sahni, Fundamentals of data structures in PASCAL, Computer Science
Press, Inc., New York, NY, 1984

12 Donald E. Knuth, The art of computer programming, volume 1 (3rd ed.): fundamental
algorithms, Addison Wesley Longman Publishing Co., Inc., Redwood City, CA, 1997

13 Harry R. Lewis , Larry Denenberg, Data Structures and Their Algorithms, Addison-Wesley
Longman Publishing Co., Inc., Boston, MA, 1997

14 D. Liles, P. Mamnami, R. Sinclair, J. Walgenbach, & S. Williams. ROD User's Guide. Class notes
for Software Development, Computer Science Dept., New Mexico State Univ. (Spring 1994).

15 J. S. Miller. MultiScheme: a Parallel Processing System Based on MIT Scheme, Ph.D. dissertation,
Mass. institute of Tech. (1987).

16 Moon, David A. MA CLISP Reference Manual, Project MAC, Mass. Institute of Tech. (April 1974).

17 Young Park , Benjamin Goldberg, Static analysis for optimizing reference counting, Information
Processing Letters, v.55 n.4, p.229-234, Aug. 25, 1995

18 William Pugh, Skip lists: a probabilistic alternative to balanced trees, Communications of the
ACM, v.33 n.6, p.668-676, June 1990

19 A. J. Perlis , Charles Thornton, Symbol manipulation by threaded lists, Communications of the
ACM, v.3 n.4, p.195-204, April 1960

20 Thomas A. Standish, Data Structure Techniques, Addison-Wesley Longman Publishing Co., Inc.,
Boston, MA, 1980

21 R. Tarjan. Finding dominators in directed graphs. SIAM J. Comput. 3, I (March 1974), 62-89.

22   David N. Turner , Philip Wadler , Christian Mossin, Once upon a type, Proceedings of the seventh international conference on Functional programming languages and computer architecture, p.1-11, June 26-28, 1995, La Jolla, California, United States

23   J. Weizenbaum, Symmetric list processor, Communications of the ACM, v.6 n.9, p.524-536, Sept. 1963

24   J. Weizenbaum, More on the reference counter method of erasing list structures, Communications of the ACM, v.7 n.1, p.38, Jan. 1964

25   David S. Wise, Referencing lists by an edge, Communications of the ACM, v.19 n.6, p.338-342, June 1976

26   David S. Wise, Design for a multiprocessing heap with on-board reference counting, Proc. of a conference on Functional programming languages and computer architecture, p.289-304, January 1985, Nancy, France

27   David S. Wise, Stop-and-copy and one-bit reference counting, Information Processing Letters, v.46 n.5, p.243-249, July 9, 1993

28   D. S. Wise, B. Heck, C. Hess, W. Hunt, and E. Ost. Uniprocessor performance of reference-counting hardware heap. Technical Report 401, Computer Science Dept., Indiana Univ. (June 1994).

↑ CITINGS

David J. Roth , David S. Wise, One-bit counts between unique and sticky, ACM SIGPLAN Notices, v.34 n.3, p.49-56, March 1999

↑ INDEX TERMS

Primary Classification:
 D. Software
  ↳ **D.3** PROGRAMMING LANGUAGES
   ↳ **D.3.3** Language Constructs and Features
    ↳ **Subjects:** Data types and structures

Additional Classification:
 B. Hardware
  ↳ **B.3** MEMORY STRUCTURES
   ↳ **B.3.2** Design Styles
    ↳ **Subjects:** Primary memory

 D. Software
  ↳ **D.3** PROGRAMMING LANGUAGES
   ↳ **D.3.3** Language Constructs and Features
    ↳ **Subjects:** Abstract data types; Dynamic storage management

 E. Data
  ↳ **E.2** DATA STORAGE REPRESENTATIONS

↳ **Subjects:** Linked representations

General Terms:
Languages

Keywords:
garbage collection, reference counting, storage management, tags

↑ **Collaborative Colleagues:**

Joshua Walgenbach: David S. Wise

David S. Wise:		
	S Kamal Abdali	Willie Hunt
	S. Kamal Abdali	Kasey N. Klipsch
	Gregory A. Alexander	Eric Ost
	Jeremy D. Frens	David M. R. Park
	Jeremy David Frens	David J. Roth
	Daniel P. Friedman	Guy L. Steele
	Yuhong Gu	Joshua Walgenbach
	Brian Heck	Mitchell Wand
	Brian C. Heck	Richard L. Wexelblat
	Caleb Hess	

↑ **This Article has also been published in:**

● **ACM SIGPLAN Notices**
Volume 31 , Issue 6 (June 1996)

Efficient process interaction with threads in parallel discrete event simulation

Full text   Pdf (81 KB)

Source   Winter Simulation Conference archive
Proceedings of the 30th conference on Winter simulation table of contents
Washington, D.C., United States
Pages: 451 - 458
Year of Publication: 1998
ISBN:0-7803-5134-7

Authors   Reuben Passqini Department of Computer Sciences, Purdue University, West Lafayette, IN
Vernon Rego   Department of Computer Sciences, Purdue University, West Lafayette, IN

Sponsors   IIE : Institute of Industrial Engineers
SCS : Society for Computer Simulation
ASA : American Statistical Association
NIST : National Institue of Standards & Technology
IEEE-CS : Computer Society
IEEE-SMCS : Systems, Man & Cybernetics Society
ACM: Association for Computing Machinery
INFORMS/CS : Computer Science TC
SIGSIM: ACM Special Interest Group on Simulation and Modeling

Publisher   IEEE Computer Society Press   Los Alamitos, CA, USA

Additional Information: references   index terms   collaborative colleagues

Tools and Actions: Discussions   Find similar Articles   Review this Article
Save this Article to a Binder   Display in BibTex Format

↑ REFERENCES

Note: OCR errors may be found in this Reference List extracted from the full text article. ACM has opted to expose the complete List rather than only correct and linked references.

1   John S. Carson, Modeling and simulation worldviews, Proceedings of the 25th conference on Winter simulation, p.18-23, December 12-15, 1993, Los Angeles, California, United States

2   David R. Jefferson, Virtual time, ACM Transactions on Programming Languages and Systems (TOPLAS), v.7 n.3, p.404-425, July 1985

3   Richard M. Fujimoto, Time warp on a shared memory multiprocessor, Transactions of the Society for Computer Simulation International, v.6 n.3, p.211-239, Jul. 1989

4   Martin, D. E., and T. J. McBrayer. 1997. Warped- a parallel discrete event simulator (documentation for version 0.8). Dept. of EECS, University of Cincinnati, OH.

5   Rajive L. Bagrodia, Iterative design of efficient simulations using Maisie, Proceedings of the 23rd

conference on Winter simulation, p.243-247, December 08-11, 1991, Phoenix, Arizona, United States

6   Fabian Gomes , John Cleary , Alan Covington , Steve Franks , Brian Unger , Zhong-e Ziao, SimKit: a high performance logical process simulation class library in C++, Proceedings of the 27th conference on Winter simulation, p.706-713, December 03-06, 1995, Arlington, Virginia, United States

7   C. J. M. Booth , D. I. Bruce, Stack-free process-oriented simulation, Proceedings of the eleventh workshop on Parallel and distributed simulation, p.182-185, June 10-13, 1997, Lockenhaus, Austria

8   Edward Mascarenhas , Felipe Knop , Vernon Rego, ParaSol: a multithreaded system for parallel simulation based on mobile threads, Proceedings of the 27th conference on Winter simulation, p.690-697, December 03-06, 1995, Arlington, Virginia, United States

9   Herb Schwetman, CSIM: a C-based process-oriented simulation language, Proceedings of the 18th conference on Winter simulation, p.387-396, December 08-10, 1986, Washington, D.C., United States

10   Steve Kleiman , Devang Shah , Bart Smaalders, Programming with threads, SunSoft Press, Mountain View, CA, 1996

11   Mascarenhas, E., and V. Rego. 1996. Ariadne: architecture of a portable threads system supporting thread migration. Software - Practice and Experience, 26:3:327- 356.

↑ INDEX TERMS

Primary Classification:
F. Theory of Computation
↳ **F.1** COMPUTATION BY ABSTRACT DEVICES
   ↳ **F.1.2** Modes of Computation
      ↳ **Subjects:** Parallelism and concurrency

Additional Classification:
I. Computing Methodologies
↳ **I.6** SIMULATION AND MODELING
   ↳ **I.6.8** Types of Simulation
      ↳ **Subjects:** Distributed; Parallel; Discrete event

General Terms:
Algorithms, Design, Measurement, Performance, Theory

↑ Collaborative Colleagues:

Reuben Passqini: Vernon Rego

Vernon Rego:		
	Ling-Yu Chuang	Reuben Pasquini
	Ke-Hsiung Chung	Reuben Passqini
	Kehsiung Chung	Janche Sang
	Bozhidar Dimitrov	V. S. Sunderam
	Juan Carlos Gomez	Vaidy Sunderam
	Felipe Knop	Vaidy S. Sunderam
	Edward Mascarenhas	Wojceich Szpankowski

Aditya Mathur   Wojciech Szpankowski
Aditya P. Mathur
Hisao Nakanishi

CiteSeer Find: |allocated thread simulates dynami| [Documents] [Citations]

Searching for **PHRASE allocated thread simulates dynamically allocated thread**.
Restrict to: Header Title Order by: Expected citations Hubs Usage Date Try: Google (CiteSeer)
Google (Web) CSB DBLP
No documents match Boolean query. Trying non-Boolean relevance query.
500 documents found. **Order: relevance to query.**

Runtime Mechanisms for Efficient Dynamic Multithreading - Karamcheti, Plevyak, Chien (1996) (Correct)
(7 citations)
caller's stack, lazily creating a different heap-**allocated thread** only if it suspends or need be
machines for programming models with dynamic **thread** creation and multithreading requires efficient
www-csag.ucsd.edu/papers/csag/external/rtperf.ps

A C Thread Library for Multiprocessors - Schwan, Forbes, Gheith.. (1991) (Correct) (6 citations)
and the location of the memory **dynamically allocated** to the **thread**. Toward this end and to allow
1 A C **Thread** Library for Multiprocessors Karsten Schwan,
ftp.cc.gatech.edu/pub/coc/tech_reports/91/GIT-CC-91-02.ps.Z

Kernel-Level Threads for Dynamic, Hard Real-Time Environments - Marty Humphrey (1995) (Correct)
(3 citations)
scheme, in which objects are not actually **allocated** until after a page fault occurs. This memory
Kernel-Level **Threads** for Dynamic, Hard Real-Time Environments Marty
counter.cs.umd.edu/~rich/courses/cmsc818G-s98/papers/spring_kernel.ps

C Threads - Coopers, Draves (1990) (Correct)
and mutex_free(provide dynamic storage **alloca**tion and deallocation. The functions mutex_init(
C **Threads** Eric C. Cooper Richard P. Draves Department of
ftp.cs.cuhk.hk/pub/mach3/doc/techreports/threads.ps

Software-Directed Register Deallocation for.. - Lo, Parekh, Eggers, .. (Correct) (4 citations)
at knowing when a new physical register must be **allocated**, they have limited knowledge of when physical
require large register files to hold multiple **thread** contexts that can issue instructions out of order
www-cse.ucsd.edu/users/tullsen/TPDS99.ps

A Copying Collector for C++ - Edelson (1991) (Correct) (1 citation)
is not responsible for freeing **dynamically allocated** storage. Many reclamation schemes improve
In the linked-list implementation, the list is **thread**ed in the runtime stack, or in global data for
allocated. A global or static object can be **simulat**ed using a global root that always references the
ftp.edelsonassoc.com/pub/ede_us91.ps.gz

Beyond Multiprocessing - Multithreading the SunOS Kernel - Eykholt, Kleiman.. (1992) (Correct) (78 citations)
of the process, which is also swappable, was **allocated** with the user structure in the user area,
challenges: symmetric multiprocessing, multi-**thread**ed applications, real-time, and multimedia, led to
sunsite.unc.edu/pub/sun-info/development-tools/multi-threaded/beyond_mp.ps

Implementing a Parallel C++ Runtime System for.. - Bodin, Beckman.. (1993) (Correct) (4 citations)
variables i.e.program static data or data **allocated** on the heap by the main control **thread**. Each
addressing mechanism to support the name space. A **thread** system 1 This research is supported by ARPA
www.cs.uoregon.edu/research/paracomp/proj/tau/../papers/sc93.ps.gz

Multi-threading and Remote Latency in Software DSMs - Thitikamol, Keleher (1997) (Correct) (16 citations)
simplifies handling the scope of heap and stack-**allocated** data. This data is usually private to each
Multi-**thread**ing and Remote Latency in Software DSMs
www.cs.umd.edu/~keleher/papers/dcs97.ps.gz

Pickling threads state in the Java system - Bouchenak (1999) (Correct) (14 citations)
J. Kim, H. Lee et S. Lee. Replicated Process **Alloca**tion for Load Distributed in Fault-Tolerant
1 Pickling **threads** state in the Java system S. Bouchenak SIRAC

between machines. It also allows classes to be **dynamically** loaded and therefore to be moved between
sirac.imag.fr/PUB/99/99-ersads-sara-PUB.ps.gz

On Designing Lightweight Threads for Substrate Software - Haines (1997)   (Correct)   (6 citations)
minimal, consisting usually of an execution stack **allocated** in heap space and the set of CPU registers, and
On Designing Lightweight **Threads** for Substrate Software Matthew Haines
www.sois.alaska.edu/CSLG_index/usenix_tech.97/PROCEEDINGS/haines.ps

Dynamic Processor Allocation with the Solaris Operating System - Yue (1998)   (Correct)   (11 citations)
Dynamic Processor **Allocation** with the Solaris TM Operating System
processor resources to the applications' parallel **threads**. Coscheduling related parallel **threads**, or
ftp-mount.ee.umn.edu/pub/faculty/lilja/papers/llpc-on-solaris.ps

A Thread Taxonomy for MPI - Skjellum, Protopopov, Hebert (1996)   (Correct)   (6 citations)
etc.The MPICH implementation internally **allocates** such handles for every operation. These handles
A **Thread** Taxonomy for MPI Anthony Skjellum, Boris
www.cs.msstate.edu/~tony/documents/Message-Passing/ThreadTaxonomy.ps.Z

Real-Time Mach: Towards a Predictable Real-Time System - Tokuda, Nakajima, Rao (1990)   (Correct)
(165 citations)
of system and task interactions (e.g.memory **allocat**ion/deallocation, message communications, I/O
In this paper, we describe a real-time **thread** model, real-time synchronization, and the ITDS
mmmc.jaist.ac.jp:8000/publications/1990/PostScript/usenix90.ps.gz

Recycling in Gardens: Efficient Memory Management for a Parallel .. - Siu Yuen   (Correct)
and Smalltalk, Mianjin supports **dynamically allocated** data structures, which requires automatic
segments. Tasks are created **dynamically**. A task's **thread** of control may terminate but objects in the
collection. Furthermore since tasks may be **dynamically** created, heap segments and tasks must also be
sky.fit.qut.edu.au/~proe/papers/PART98.ps.gz

Mach Threads and the Unix Kernel: The Battle for Control - Tevanian, Jr., Rashid.. (1987)   (Correct)
(34 citations)
facility. For example, in Dynix [4] users can **allocate** a number of processes equal to the number of
Mach **Threads** and the Unix Kernel: The Battle for Control
www.ee.umd.edu/courses/enee647/threads/published.threads87.ps

Matisse: A system-on-chip design methodology.. - Verkest, Jr.. (1999)   (Correct)   (5 citations)
algorithms that operate on large, **dynamically allocated**, stored data structures (e.g. linked lists,
their own local virtual memory space and default **thread** of control. They are only created at
by complex algorithms that operate on large, **dynamically allocated**, stored data structures (e.g. linked
imecgate.imec.be/vsdm/projects/matisse/../../ftp/pub/iwv98.ps.gz

Evolving Mach 3.0 to a Migrating Thread Model - Ford, Lepreau (1994)   (Correct)
migrating RPC is made into the server, the kernel **allocates** an unoccupied activation from the server's
Evolving Mach 3.0 to a Migrating **Thread** Model Bryan Ford Jay Lepreau University of Utah
ftp.cs.utah.edu/pub/thread-migrate.ps.Z

A Fast Parallel Conservative Garbage Collector for Concurrent.. - Matsuoka, al. (1991)   (Correct)   (3 citations)
scheme that meet the demands of very fast **allocat**ion -up to one million objects per second. Our
for multitudes of different languages. Lightweight **thread** is one popular abstraction recently, more
ftp.yl.is.s.u-tokyo.ac.jp/pub/papers/woos91-gc-a4.ps.Z

First 20 documents Next 20

Try your query at: Google (CiteSeer)   Google (Web)   CSB   DBLP